ARM7を使って計測・制御システムを作る

——A-D/D-Aコンバータのディスコンに対応するテクニック

2006年3月号付属ARM基板関連企画

システムの記事　関連の記事　関連データ

上田智章

生産数量の少ない計測・制御システムの中には，製品寿命が5～10年と長いものが珍しくない．ところが，これらが内蔵しているA-DコンバータやD-Aコンバータは数年で出荷中止になる場合もある．ここでは本誌2006年3月号に付属したARM基板が搭載しているADuC7026（米国Analog Devices社製）の汎用I/Oポートを使い，A-DコンバータやD-Aコンバータを構成するテクニックを紹介する．（編集部）

計測・制御システムは，長期に渡って運用されるものが大半を占めることから，製品寿命が5～10年と長いものが少なくありません．ところが，プリント基板開発時に選定したA-DコンバータやD-Aコンバータが数年で製造中止になり，慌てるケースをよく耳にします．これらの部品には互換性のないものが多いからです．結局，プリント基板を再設計したり，ソフトウェアを作り直すことになります．

また，マイコンに内蔵されているA-DコンバータやD-Aコンバータのチャネル数やビット分解能が不足したとき，上位チップへの置き換えや外付けICの追加で対応しようとすると，コスト的に見合わないというケースもあります．さらに，高機能化に伴って測定レンジごとに振幅やオフセットの微調整が入り，マルチプレクサや電子ボリュームが追加されて，電子回路が複雑になるケースもあると思います．

本稿では，ARM7コアの汎用ディジタルI/Oポートを使って，簡易な外付け回路だけで任意の分解能をもつD-Aコンバータを構成する手法を紹介します．そしてこれをA-Dコンバータに応用する方法を説明します．後半では，電子ボリュームやオフセット調整機能を組み込んだDDS（direct digital synthesizer）を利用して三相交流を作ります．

● アナログ部品が製造中止になると開発の苦労が水の泡

アナログ回路を搭載するシステムでは，**図1**に示すように仕様や開発目標に対する部品の選定だけでもたいへんです．A-D/D-Aコンバータのチャネル数やビット分解能，精度，ピン配置，メモリ容量，基板面積，故障率などの要求項目をクリアして開発しても，肝心の部品が数年で製造中止になることがあります．とくに外付けのA-D/D-Aコンバータが問題となることが多いようです．また，1枚数万円のボードに数千円もするA-D/D-Aコンバータを搭載するのはコスト的にも厳しいと言えます．

● 汎用I/Oポートを利用してD-Aコンバータを作る

汎用ディジタル・ポートをD-Aコンバータとして使う方法と言えば，多くの方は**図1(c)**に示すようなラダー抵抗を使った簡易なD-Aコンバータや，**図1(d)**に示すPWM（pulse width modulation；パルス幅変調）を使ったD-Aコンバータを思い浮かべることでしょう．確かに運良くこうした簡易なD-Aコンバータを採用できるケースはあります．

1）ラダー抵抗を使う簡易なD-Aコンバータ

図2(a)に示すようなラダー抵抗を用いたD-Aコンバータの場合，多数のI/Oポートが必要です．精度的にはディスクリートなら8ビット，専用ラダー抵抗アレイを使っても10ビットが限界でしょう．

2）PWM出力を使うD-Aコンバータ

図2(b)に示すように，PWM出力がある場合には，出力

KeyWord　ADuC7026，ARM7コア，D-Aコンバータ，ラダー抵抗，PWM，PDM，ARM基板，DDS，逐次比較型，Σ-Δ型，三相交流

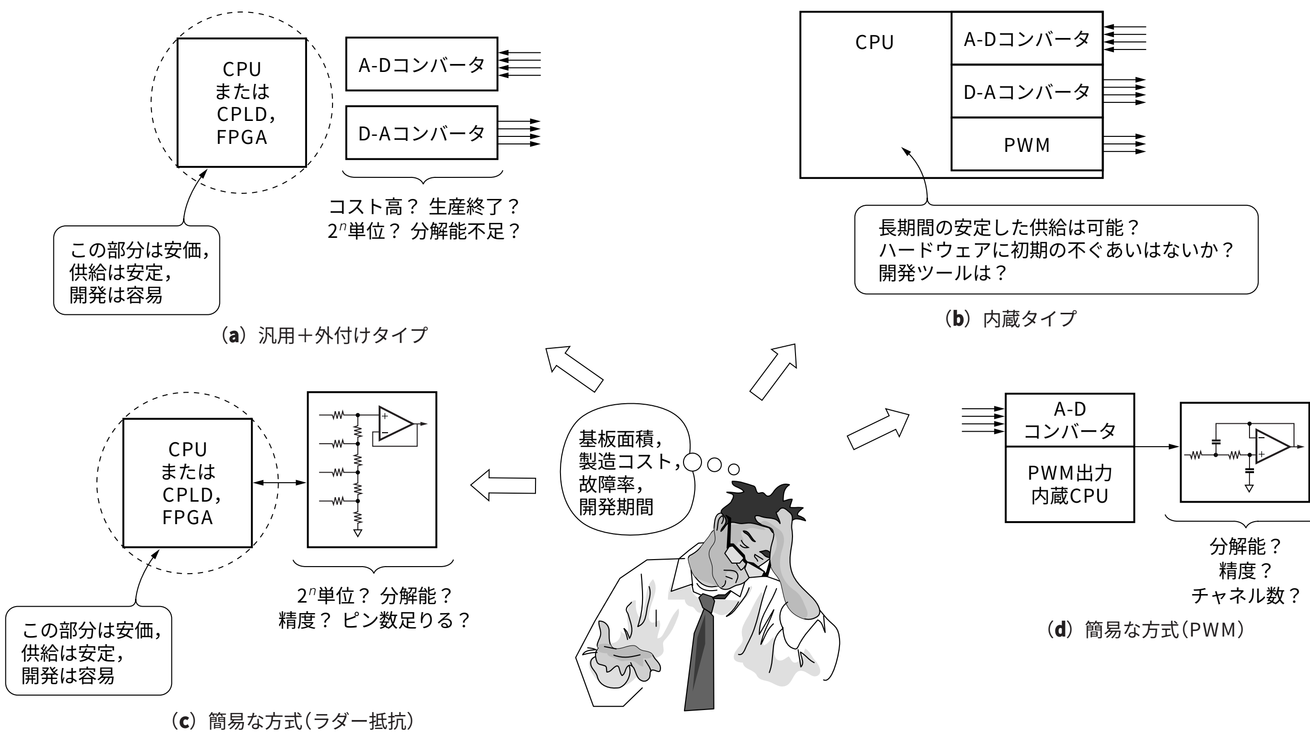


図1　A-D/D-Aコンバータは数年で製造中止になることがあり，設計者の悩みは尽きない
採用に際しては，チャネル数やビット分解能，精度，ピン配置，メモリ容量，基板面積，故障率など，考慮すべき点がたくさんある

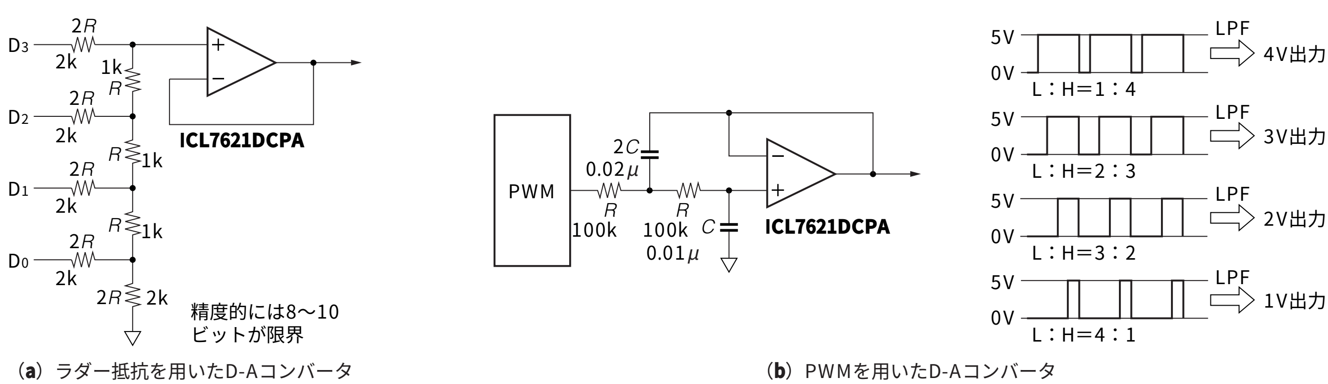


図2　汎用ディジタル・ポートをD-Aコンバータとして使う場合，たいていの人が思いつく方法
運が良ければ目的にあったものを開発できる

にローパス・フィルタ(LPF：low pass filter)を入れて直流成分だけを取り出して，D-Aコンバータとして使えます．しかし，たいていは分解能が10ビット程度ですし，チャネル数も限られます．モータを制御する基板なら，すでにそのポートは使われていることでしょう．

3) PDM型D-Aコンバータ

図3にPDM(pulse density modulation；パルス密度変調)型D-Aコンバータのアルゴリズムを例示します．このアルゴリズムには加減算程度しか必要がないので，ソフト

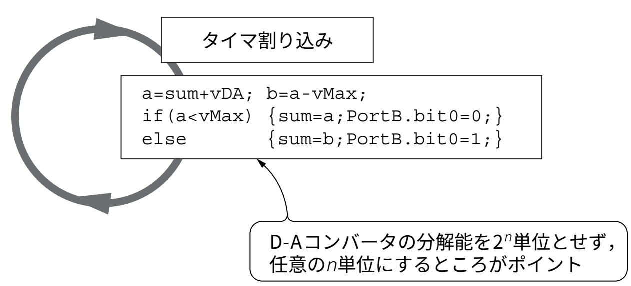


図3　PDM型D-Aコンバータのアルゴリズム
このアルゴリズムには加減算程度しか必要ないため，ソフトウェアでもハードウェアでも実行可能

ウェアでもハードウェアでも実行できます．

ソフトウェアの場合には，タイマ割り込み中に実行します．変数`vMax`で分解能を任意の自然数nに指定できます．D-Aコンバータの出力値は変数`vDA`で指定します．変数`sum`はパルスを発生するか否かを判断するキャリ・オーバを蓄積するために使います．このPDM型D-Aコンバータの分解能は2のn乗単位ではなく，$n+1$階調の出力（0からnまで）を実現します．例えば1mVステップで出力を指定できます．

図4(a)のように，ほぼ直流出力に近いケースなら抵抗とコンデンサ，保護ダイオードの3個で構成されるローパス・フィルタで十分です．I/Oポートにはパルス信号が出力されますが，簡易なフィルタでも信号成分を取り出せます．

図4(b)のようにOPアンプでローパス・フィルタを構成すれば，帯域幅をいくぶん広げることができます．このローパス・フィルタは出力パルスの高周波ノイズ成分をカットし，*SN*比を改善するためのものなので，シャープな特性のフィルタほどきれいな出力が得られます．

● PWM方式 vs. PDM方式

図5(a)に示すように，PWMはパルスの周期が固定され，‘1’と‘0’のデューティ比が設定値に応じて変化する変調方式です．米国Analog Devices社のADuC7026（ARM7TDMIコアを内蔵）のPWMには16ビットの分解能があり，クロックが41.78MHzのときに24ns単位で時間増分を変えられますが，設定値の変更は318.75サンプル/s程度になります．このPWMで60Hzの信号を作ろうとすると，正弦波1周期当たり5サンプル程度しかないため，*SN*比が悪くなってしまいます．

これに対してPDMは，1クロック期間には‘0’か‘1’のパルスが1個だけ入り，パルスの出現頻度を確率的に決定する方式です（**図5(b)**）．「確率的に」などと言うと何やら難しそうですが，**図3**のアルゴリズムで変数`sum`に毎回設定値`vDA`を累積加算し，設定最大値`vMax`を越えるキャリ・オーバが発生するタイミングでパルス‘1’を与えるだけの簡単な計算で済む，という特徴があります．

ADuC7026では，タイマ割り込みの周期を15μsとした場合，信号帯域が60Hz程度ならば，周期クロックと情報帯域が十分に離れています．そのため，上の周波数成分に含まれるノイズ成分を，**図4**に示したような簡単なローパス・フィルタでカットできます．

● PDM型D-Aコンバータの特徴

図4(a)に示すように，抵抗とコンデンサ，保護ダイオードだけで構成したローパス・フィルタを使ってPDM型D-Aコンバータの直線性を調べた結果を**図6**に示します．**リスト1**に5μsごとに実行したアルゴリズムを示します．3.3V電源で1mVの分解能を実現しようとしたので，`vMax=3300`，`vOFS=0`（オフセットはなし）とし，`vDA`を

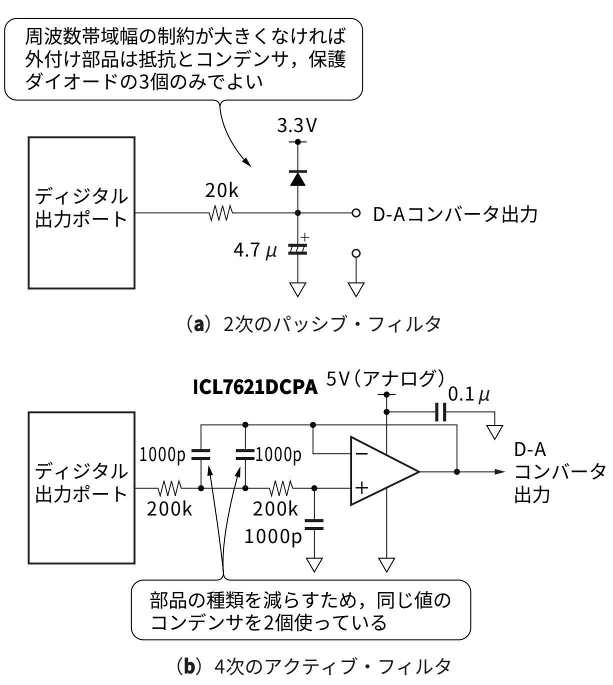


図4　PDM型D-Aコンバータに必要な外部回路の例
シャープな特性のフィルタほどきれいな出力が得られる

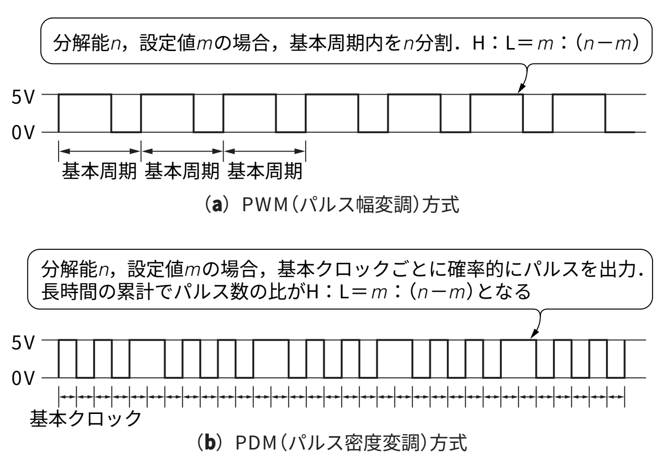


図5　PWM方式とPDM方式の違い
PWMはパルスの周期が固定され，‘1’と‘0’のデューティ比が設定値に応じて変化する変調方式．PDMは1クロック期間には‘0’か‘1’のパルスが1個だけ入り，確率的にパルスの出現頻度を決定する方式

変えながらD-Aコンバータの出力電圧を測定しました．この結果，実際の回路ではvMax=3300（‘1’のパルスが連続で出力される状態）では3.290Vにしかならないことがわかりました．そこで，vMax=3290と補正して再度測定すると，**図6**に示すようにほぼ1mVステップで電圧を出力できるようになりました．

PDM型D-Aコンバータはオフセット調整を簡単に行えます．**リスト1**の中のvOFSを0以外の値に変更するだけで良いのです．

複数の測定ダイナミック・レンジを持った測定装置の場合には，チャネルごとにオフセットだけでなく，倍率（感度，ゲイン）変更の必要が生じるケースがあります．このような場合，回路は複雑になります．ところが，PDM型D-Aコンバータでは，変数vMaxの値を変更するだけで倍率を変更できます．vMaxを通常の1/*n*にすれば倍率は*n*倍に，反対に*n*倍にすれば倍率は1/*n*倍になります．**図6**を参考にしてください．

● PDM型DACを使ってA-Dコンバータを実現

最近はアナログ・コンパレータを実装したマイコンもあるため，D-Aコンバータが実現できれば，A-Dコンバータも実現できます．

図7(a)に示すように，D-Aコンバータにリファレンス信号を出力させ，アナログ・コンパレータと組み合わせると逐次比較型A-Dコンバータを構成できます．

積分器で長時間の変換誤差が0になるようにフィードバックするΣ-Δ型A-Dコンバータで利用されている方法などを使って，**図7(b)**に示すようなA-D変換を行えます．

● 本誌付属基板でPDM型D-Aコンバータを実現する

本誌2006年3月号の付属基板に搭載されていたプロセッサはADuC7026でした．このチップは内蔵クロック41.78 MHzで動作でき，A-Dコンバータ，D-Aコンバータが実装

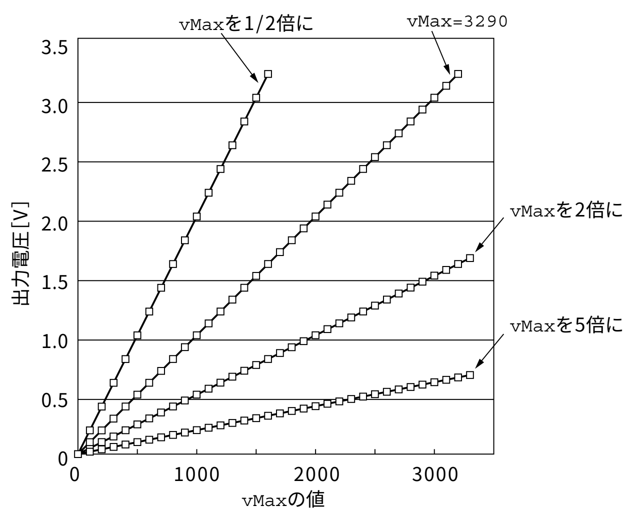
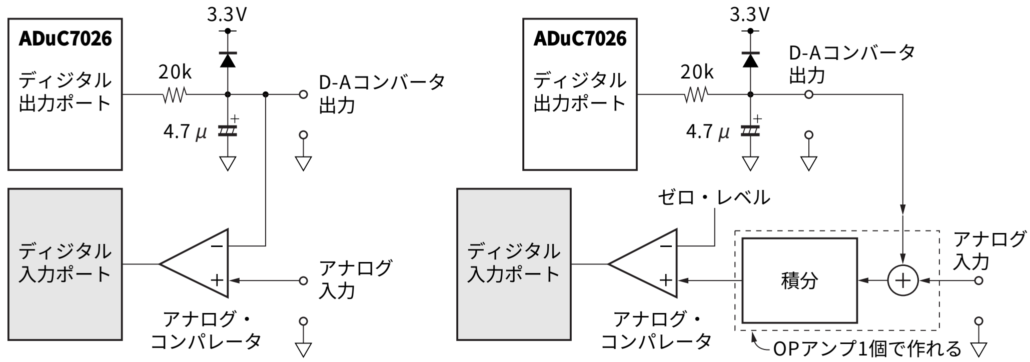


図6　PDM型D-Aコンバータの直線性を調べた結果
ほぼ1mVステップで電圧を出力できた

リスト1　PDM型D-Aコンバータはオフセット調整を簡単に行える

```
if (flagIRQ==1)           //flagIRQが1ならタイマ割り込みがあったことを示す
{  vSum += vDA + vOFS;
   PortData=0xFF000000
   if ( vSum > vMax )[ PortData |=0x00C10000; vSum-=vMax; ] eles PortData|=0x00080000;
GP4DAT=PortData;
flagIRQ = 0;              // タイマ割り込みフラグをクリアする
}
```



(a) 逐次比較型A-Dコンバータ　(b) 高精度型A-Dコンバータ

図7　PDM型D-Aコンバータを使ったA-Dコンバータの実現方法
最近はアナログ・コンパレータを実装したワンチップ・マイコンもあるため，D-Aコンバータさえ実現できれば，A-Dコンバータを実現できる

されているため，単体でも魅力のあるデバイスです．このARM基板を使って，PDM型D-AコンバータによるDDSを実現し，内蔵D-Aコンバータとの比較を行ってみました．

1）ADuC7026に内蔵するD-Aコンバータの出力

内蔵のD-Aコンバータを使って付属基板単体でDDSを構成しました．一般的なDDSは**図8(a)**のような構成になっています．コア・クロック41.78MHzで動作させ，タイマ割り込み周期を約5μsとし，サイズ512の配列に1波の正弦波を定義しています．

内蔵D-Aコンバータを使ったところ，最大10kHzまで応答するDDSをソフトウェアで実現できました．**図8(b)**に出力波形を示します．よく見ると階段状になっていることがわかります．また，出力範囲が0V～3.3Vに固定されており，振幅を調整するには外部に電子ボリュームを追加するか，波形メモリ(配列)内のデータを変更する必要があります．

図9に内蔵D-Aコンバータを使った3相DDSの構成図を示します．演算負荷を低減するため，120 °ずつずれた正弦波を格納したサイズ512の配列を三つ用意しています．ちなみに米国Microchip Technology社のPICマイコンではメモリ・サイズの関係から，サイズ128の配列1個が定義できる程度です．

2）PDM型D-Aコンバータの出力

図9の出力段のD-AコンバータをPDM型D-Aコンバータに変えるだけで，PDM型DDSになります(p.116の**図10(a)**)．PDM型D-Aコンバータは符号を反転するだけで180 °位相の違う信号を簡単に作ることができます．p.116の**リスト2**にプログラムを示します．

内蔵D-Aコンバータは0V～3.3Vの範囲に固定した信号しか出力できませんが，PDM型は振幅とオフセットを変更し，6相の信号を出力しています．**図10(b)**にPDM型D-Aコンバータの出力波形を例示します．この波形は30Hz設定時にOPアンプを使ったローパス・フィルタ出力を観測したものです．

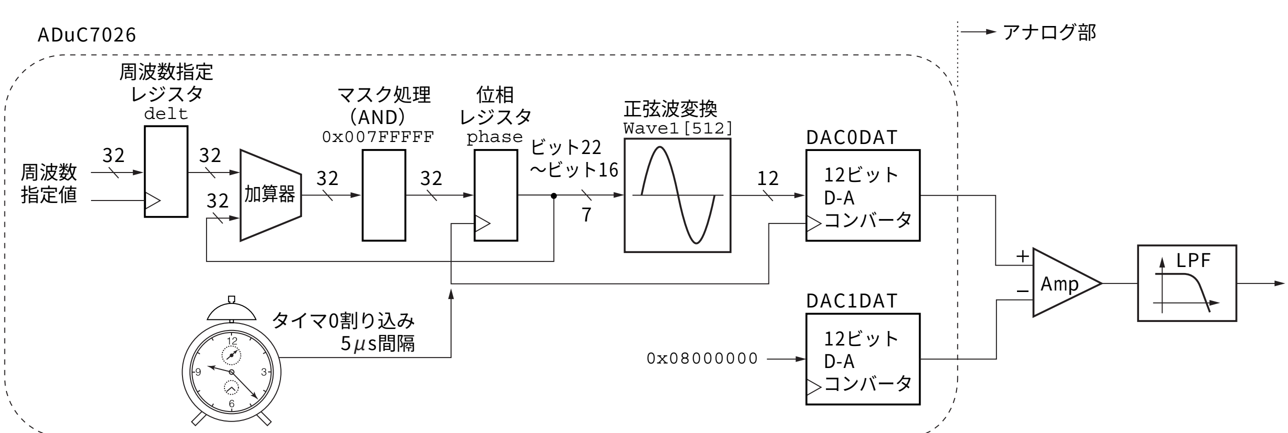
● いったいどこまでの性能が出るのか？

写真1にDDSの出力波形とローパス・フィルタ通過後の波形を示します．筆者の実験では，**図4(b)**に指定した回路定数ならば，60Hz程度まではソフトウェアだけで実現できるという結果を得ました(**図11**)．ローパス・フィルタを通さず，PDM出力をそのまま使えば，モータ制御では3600rpmに対応できるので，応用範囲は広そうです．

このアルゴリズムは加減算程度しか使わないので，ハードウェアでも容易に実行できます．ソフトウェアだけの場合に比べて動作速度を数百倍引き上げることができるでしょう．詳細は別の機会に紹介したいと思います．

＊　　＊

本稿では，任意の分解能を持つPDM型D-Aコンバータの実現方法を提案し，本誌2006年3月号付属のARM基板を使って振幅とオフセットの調整機能のある三相DDSを構成しました．通常のD-Aコンバータでありがちな階段状の出力波形にならず，比較的滑らかな波形を確認できました．ソフトウェアだけでは動作速度に制約があるため応用範囲



(a) ブロック図

図8　ADuC7026に内蔵するD-Aコンバータで作るDDS
最大10kHzまで応答するDDSをソフトウェアで実現した

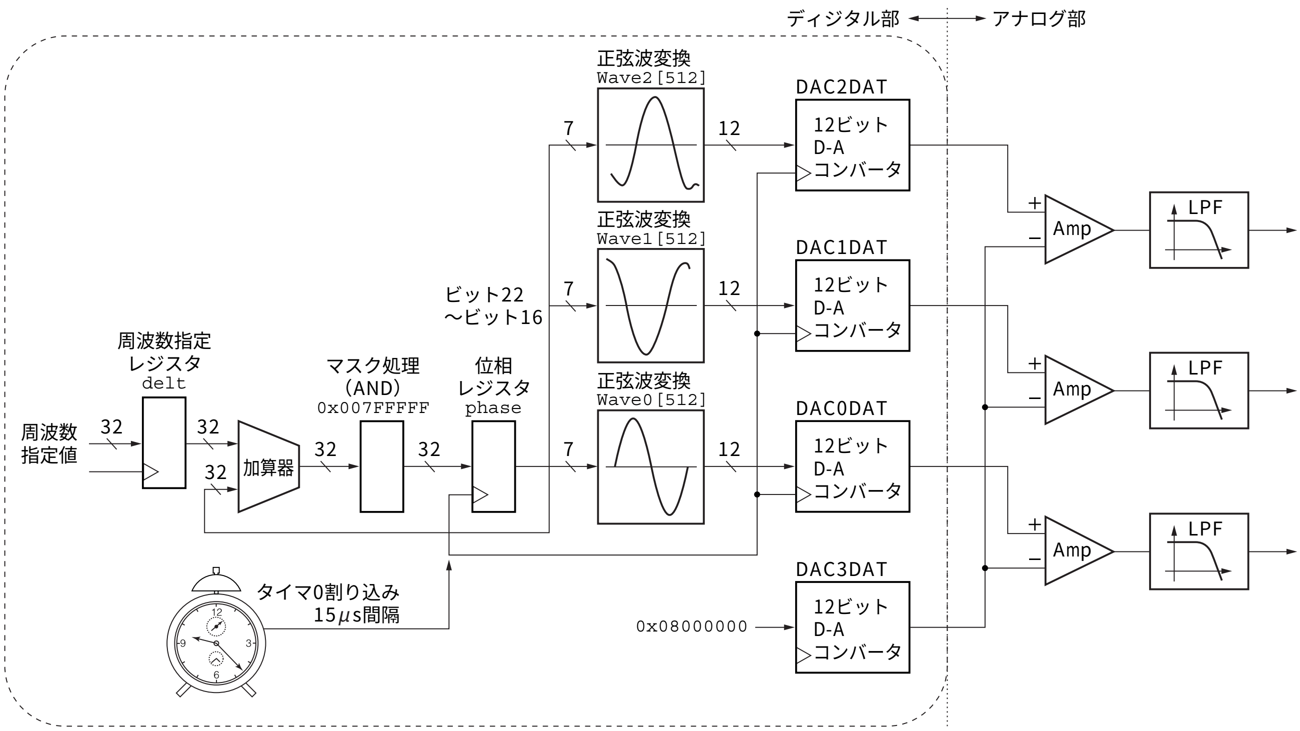


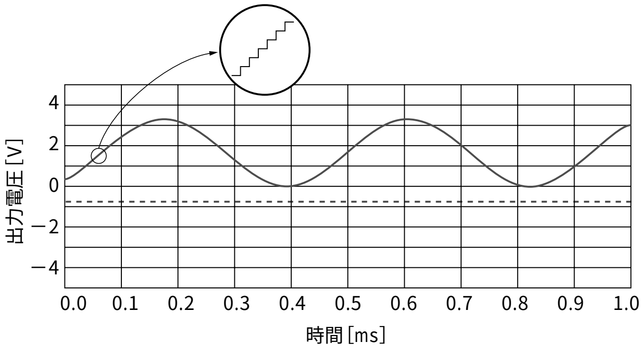
図9　ADuC7026が内蔵するD-Aコンバータを使った3相DDS
演算負荷を低減するため，120 °ずつずれた正弦波を格納したサイズ512の配列を三つ用意している

は限られるかもしれませんが，汎用ディジタル出力しかなくてもA-DコンバータやD-Aコンバータを安価に構成できることが示せたと思います．

PDM型D-Aコンバータのように2値の符号を扱う回路の乗算処理は，排他的論理和XORで実現できます．つまり，同期検波回路やインピーダンス・アナライザなどの測定器の機能も組み込める可能性を秘めています．さらに，FPGAやCPLDに対してPDM型のD-AコンバータやA-Dコンバータの組み込みが可能であるため，アナログ・ディジタル混在の計測・制御応用分野においても役立つのではないかと思います．

本記事で使用したC言語のソース・リストは，本誌のWebサイト（http://www.cqpub.co.jp/dwm/）と筆者のWebサイト（http://www.neo-tech-lab.com/）からダウンロードできます．追加の情報が発生した場合にも，筆者のWebサイトで公開する予定です．

うえだ・ともあき
(株)KRI センシング技術部 主席研究員



（**b**）出力波形

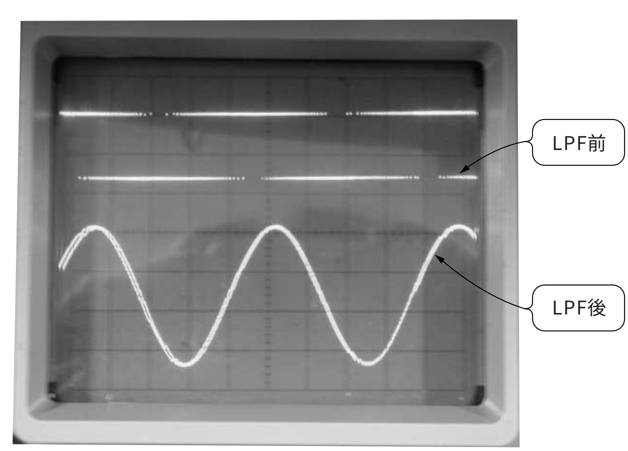


写真1　製作したPDM型D-Aコンバータの出力

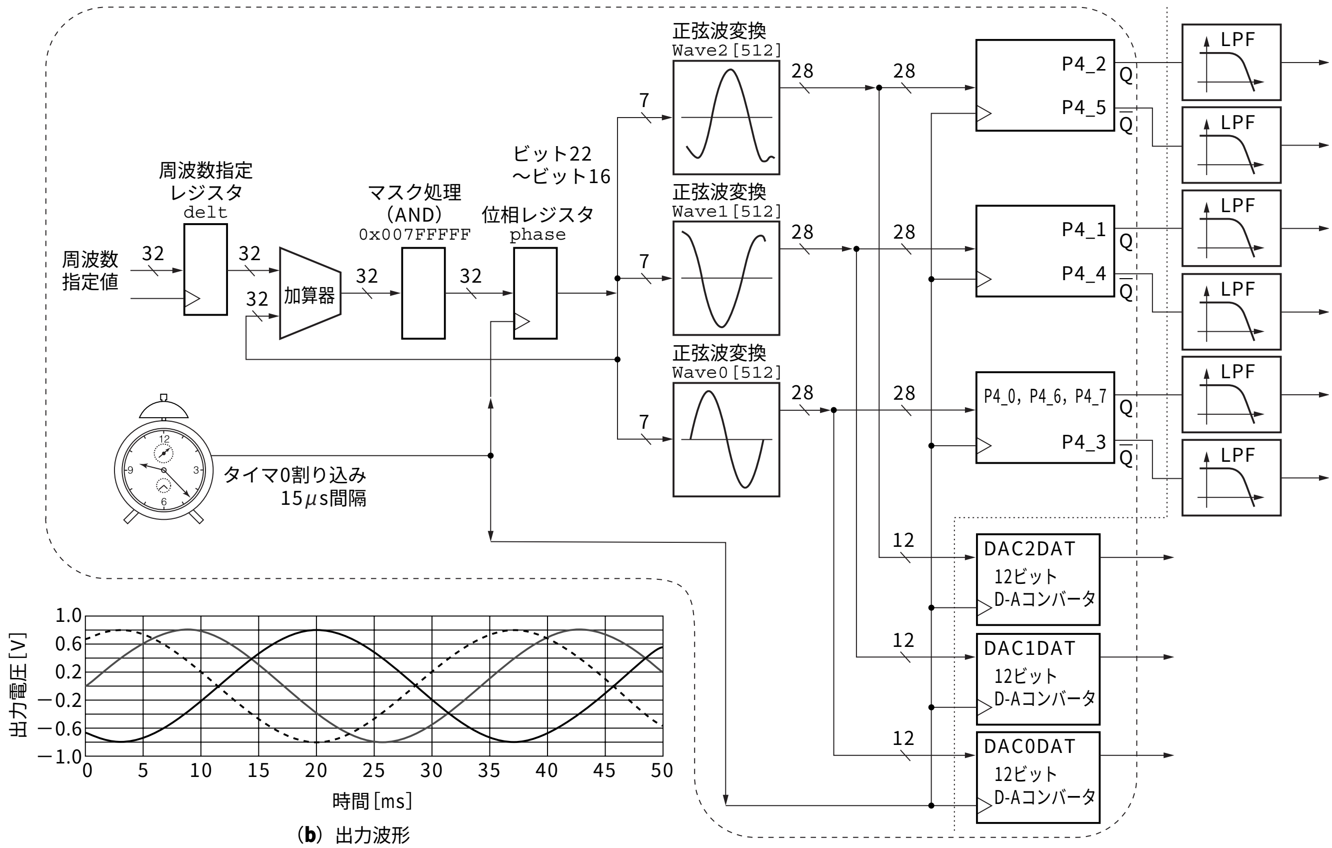


図10　出力段のD-AコンバータをPDM型D-Aコンバータに変えたPDM型DDS
符号を反転するだけで180°位相の違う信号を簡単に作ることができる

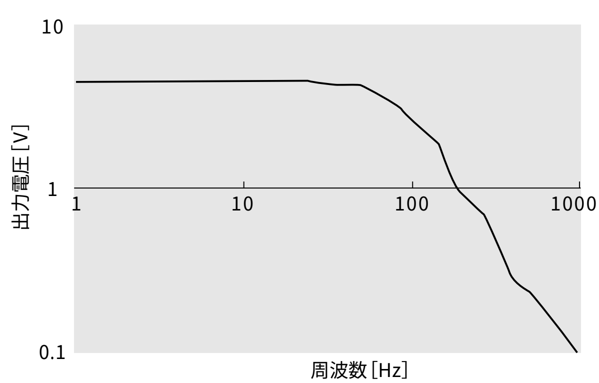


図11
PDM型D-AコンバータによるDDS出力の周波数特性例
図4(b)に指定した回路定数ならば，60Hz程度まではソフトウェアだけで実現できる

リスト2　ADuC7026で3相DDSを実現するためのプログラム
振幅とオフセットを変更し，6相出力を実現している

```
/**************************************************************************
デザインウェーブマガジン ADuC7026ボード
 任意分解能を持つパルス密度変調型D-Aコンバータ  By KEIL uVISION3
**************************************************************************/

#include <ADuC7026.H>   // ADuC7026 レジスタ定義ファイル

// 通常のD-Aコンバータ(12ビット)で出力することを前提とした正弦波データ[ステップ数512]
// 通常のD-Aコンバータが使用する部分は上位16ビットの下位けた12ビット部分のみ．下位の16ビットはPDM型D-Aでは使う．
unsigned int wave0[512] = {
   0x04000000,0x040C8DC5,0x04191B0E,0x0425A75F,0x0432323C,0x043EBB2A,0x044B41AD,0x0457C549,

       (この部分はデータなので省略)
```

```
   0x039BBA7D,0x03A83AB7,0x03B4BE53,0x03C144D6,0x03CDCDC4,0x03DA58A1,0x03E6E4F2,0x03F3723B  };
unsigned int wave1[512] = {
   0x0775F1A9,0x076F99B3,0x07691FD3,0x0762844B,0x075BC75A,0x0754E943,0x074DEA4B,0x0746CAB6,

       (この部分はデータなので省略)

   0x07A3D04E,0x079E9000,0x07992BF8,0x0793A46E,0x078DF996,0x07882BA9,0x07823AE1,0x077C2778  };
unsigned int wave2[512] = {
   0x008A0E57,0x0083D888,0x007DC51F,0x0077D457,0x0072066A,0x006C5B92,0x0066D408,0x00617000,

       (この部分はデータなので省略)

   0x00C07535,0x00B9354A,0x00B215B5,0x00AB16BD,0x00A438A6,0x009D7BB5,0x0096E02D,0x0090664D  };
```

三つの配列 wave0[], wave0[], wave0[], にはあらかじめ120°ずつ位相がずれた正弦波形が登録されている

```
unsigned int vMax, vOFS;          // PDM型D-Aコンバータの最大値と出力オフセット
unsigned int vMax0,vMax1,vMax2; // PDM型D-Aコンバータの最大値(D-Aコンバータの出力範囲は0からvMax*まで)
unsigned int vSum0,vSum1,vSum2; // PDM型D-Aコンバータの累積加算値
unsigned int  vDA0, vDA1, vDA2; // PDM型D-Aコンバータの出力値
unsigned int PortData;           // PDM型D-Aコンバータが汎用I/OポートGP4DATに書き込むデータ．すべてのビットは出力として使用
```

vMax，vOFSともに各相独立した値を指定することができる

```
int flagIRQ;          // タイマ割り込み0が発生したことを知らせるためのフラグ(1:発生)

double frequency;     // DDSが出力する周波数[単位 Hz]
int delt;             // DDS増分レジスタ(タイマ割り込みごとに位相レジスタに加算する．出力周波数に関係)
int phase;            // DDS位相レジスタ(下位16ビットは小数点以下と考え，上位16ビットの下位けた9ビット分を
                      // 配列wave0[], wave1[], wave2[]にアクセスするために使う．)

void IRQ_Handler(void) __irq{
  flagIRQ = 1; // タイマ割り込み0の発生をセットする
  T0CLRI = 1;  // ADuC7026のハードウェア割り込みフラグをクリアし，次のタイマ割り込みを許可
}
```

この予約関数がタイマ割り込みのほんとうの実行部分．フラグをセットする処理のみを行う

```
int main (void){
    int i;            // 正弦波形データ配列引き数[0から511まで]

    flagIRQ = 0; T0LD = 42*15; T0CON = 0xC0; // タイマ割り込み関連初期化処理．クロックを約42MHzと考え，42×15
                                             // クロックごとに1回タイマ0割り込みを発生するように指定している

    PLLKEY1 = 0xAA; PLLCON = 33; PLLKEY2 = 0x55; // bit5:OSEL=1, bit1_0:MDCLK=01
    POWKEY1 = 0x01; POWCON = 0;  POWKEY2 = 0xF4; // PC=active, CD=41.78MHz
```

クロックを内蔵41.78MHzに切り替えるおまじない

```
    PortData = 0xFF000000; GP4CON=0x00000000; GP4DAT=PortData; // 汎用I/O[GP4]のすべてのビットはディジタル出力

    DAC0CON = 19; DAC0DAT=0x0400FFFF; // 内蔵DAC0で中央値を出力 bit5:DACCLK=0,bit4:DACCLR=1,bit1_0:DAC_range11
    DAC1CON = 19; DAC1DAT=0x0400FFFF; // 内蔵DAC1で中央値を出力 bit5:DACCLK=0,bit4:DACCLR=1,bit1_0:DAC_range11
    DAC2CON = 19; DAC2DAT=0x0400FFFF; // 内蔵DAC2で中央値を出力 bit5:DACCLK=0,bit4:DACCLR=1,bit1_0:DAC_range11

    // PDM型D-Aコンバータの初期化処理
    // vMax=0x20000000; vOFS=0x08000000;
    // vMax=0x20000000; vOFS=0x08000000;
    vMax=0x10000000; vOFS=0x04000000;          // ここで倍率(振幅)とオフセットを指定している．
    vMax0=vMax; vSum0=0; vDA0=0;   vMax1=vMax; vSum1=0; vDA1=0;   vMax2=vMax; vSum2=0; vDA2=0;

    // DDSの初期化処理
    phase=0; // 位相レジスタを0クリアする
    frequency = 30.0;  // ここで指定した周波数[Hz]が出力される．数Hz以下の値を指定すればLEDの点滅も観測できる
    // 増分値 = (CPUのクロック*位相レジスタ最大値)÷(周波数 * タイマ割り込み周期カウント値)
    delt = (int)( (double)(0x1FFFFFF) *(double)(42*15)*frequency  / (double)(41780000.0) );

    IRQEN = 0x00000004; /* Timer0割り込み許可 */

    while(1)
    {  if (flagIRQ==1) // flagIRQが1ならタイマ割り込みがあったことを示すのでDDS処理とPDM処理を行う
       {  //GP4SET = 0x00800000;  // 一連のタイマ割り込み処理の処理時間を計測するために用いたコード ここから⇒
          // 通常のDDSアルゴリズム部分
          phase = (phase + delt) & 0x1FFFFFF;  // 位相レジスタを更新 (wave[512]用)
          i=(phase >> 16); // phaseレジスタの有効ビット(Xの部分)を取り出す 0000000XXXXXXXXX****************B
          vDA0 = wave0[i]; vDA1 = wave1[i]; vDA2 = wave2[i]; // 各相の正弦波の波形データを配列から取り出す
          DAC0DAT = vDA0;  DAC1DAT = vDA1;  DAC2DAT = vDA2;  // 各相のデータをそれぞれのD-Aコンバータに書き込む

          // PDM型D-Aコンバータの処理部分      (LED(bit7)とbit6に関してはbit0と同じ波形を出力させている)
          vSum0 += vDA0 + vOFS;   vSum1 += vDA1 + vOFS;   vSum2 += vDA2 + vOFS;
          PortData=0xFF000000;  // 最上位の8ビット(FF)は全ビットを出力指定するためのもの
          if ( vSum0 > vMax0 ) { PortData |= 0x00C10000; vSum0-=vMax0; } else PortData |= 0x00080000;
          if ( vSum1 > vMax1 ) { PortData |= 0x00100000; vSum1-=vMax1; } else PortData |= 0x00020000;
          if ( vSum2 > vMax2 ) { PortData |= 0x00200000; vSum2-=vMax2; } else PortData |= 0x00040000;
          GP4DAT=PortData; // 汎用I/OポートにPDM2値データを出力

          flagIRQ = 0;     // 以上の処理でタイマ割り込みを受け付けたのでフラグを消去しておく
          //GP4CLR = 0x00800000;  // 一連のタイマ割り込み処理の処理時間を計測するために用いたコード ⇒ここまで
       }
    }
}
```